Practical Exercise 9

Ομάδα : 3

Συμμετέχοντες : Μιχάλης Μιχαήλ

Σώτος Βασιλείου

Πασιουρτίδης Κώστας

A person holding a card

Description automatically generated with low confidence

Άσκηση 1 – Prim

Text

Description automatically generated with medium confidenceΣε αυτή την άσκηση αρχικά διαβάζουμε από αρχείο και αρχικοποιούμε τον γράφον όπως ακριβώς κάναμε στο assignment3 . Στο Prim μας ενδιαφέρει να δημιουργήσουμε το μονοπάτι με το λιγότερο κόστος κατασκευής. Δηλαδή κάθε φορά ψάχνουμε να δούμε από τους κόμβους που επισκεφθήκαμε ποιο είναι το μονοπάτι ( προς μη επισκέψιμο μέχρι τώρα κόμβο) με το λιγότερο βάρος και το επισκεπτόμαστε. Για την υλοποίηση ορίζουμε αρχικό κόμβο, ένα πίνακα με τους επισκεπτόμενους κόμβους και counter για τον αριθμό των διαδρομών που δημιουργήσαμε .Σε κάθε φάση σύγκρισης για τα μονοπάτια εκτυπώνουμε το μονοπάτι που επιλέγουμε να ακολουθήσουμε και προσθέτουμε το συγκεκριμένο βάρος στο ολικό.

Main (){

graph\_init\_from\_file;

//prim

Arxikos\_komvos=0 , vt[9]={} , et=0 ;

for(i=0;i<g.N;i++){ //vriski tin mikroteri diadromi apo ton arxiko komvo

if(g.array[arxikos\_komvos][i] !=0 &&( g.array[arxikos\_komvos][i] < smaller[2] ) ){

smaller[]=arxikos\_komvos , i , g.array[arxikos\_komvos][i];

}

}

for (i=1;i<(g.N-1);i++){//mexri ton aritho ton diadromwn

for(k=0;k<g.N;k++){

for (j=0 ; j<g.N ; j++){

if((vt[j]==0)&&vt[k]==1&&g.array[k][j]!=0 &&(g.array[k][j] < smaller[2] )){

smaller[]= {k,j,g.array[k][j]}

}

}

}

//vrika pia einai epomeni mikroteri diadromi

et++;

vt[smaller[1]]=1;//vrika ton epomeno komvo

mst=mst+smaller[2];

smaller[]={0,0,999999};

}

Printf(“MST of prim ”, mst);

Άσκηση 2

*Αρχικα:*

Στην main ανοιγω το αρχειο μου παιρνω τον αριθμο τον κομβων και τις ακμες και δημιουργω τον γραφο μου. Στη συνεχεια με ένα while loop παιρνω γραμμη γραμμη τις ακμες και το βαρος.

A picture containing text

Description automatically generatedSrc=η αρχη της ακμης και dest= το τελος της, Weight = βαρος.

Μετα εκτελειται η κυριως συναρτηση KruskalMST του γραφου μου.

ifstream fin

fin.open("ECE325\_undirected\_graph.txt")

int nodes,akmes,kor1,kor2,weight,i=0

fin>>nodes

fin>>akmes

Graph\* graph = createGraph(nodes, akmes)

while (!fin.eof()) {

fin>>kor1>>kor2>>weight

graph->edge[i].src = kor1

graph->edge[i].dest = kor2

graph->edge[i].weight = weight

i++ }

*Η ιδεα της συναρτησης μου αλλα και γενικα του MSΤ είναι η εξης:*

Κυρίως συναρτηση KruskalMST:

Ταξινομω όλες τις άκμες με αυξουσα σειρά βαση του βάρους τους. (qsort)

Διαλέγω την μικρότερη άκμη. Ελέγχω αν σχηματίζει κύκλο με το Spanning Tree που σχηματιζεται μέχρι τώρα. Αν δεν σχηματίζεται κύκλος, βαζω και αυτήν την άκμη. Αλλιως την απορριπτω. Επαναλαμβανω το βήμα 2 μεχρι να υπάρχουν ακμες V-1 στο Spanning Tree μου.

*Εξηγω αναλυτικα ξανα την συναρτηση σε σχολια*

KruskalMST( graph)

{

int V = graph->V

Edge result[V] // result gia MST

int e = 0

int i = 0

//sorting pinaka apo akmes se avksousa vasi ta varoi tous

qsort(graph->edge, graph->E, sizeof(graph->edge[0]),

myComp)

// Allocate memory gia na dimiourgiso V subsets

subset\* subsets = new subset[(V \* sizeof(subset))]

// dimiourgo V subsets

for (int v = 0; v < V; ++v)

{

subsets[v].parent = v

subsets[v].rank = 0

}

//arithmos akmwn pou tha paroume mexri V-1

while (e < V - 1 && i < graph->E)

{

// epilego tin mirkoteri akmi kai avksano to index gia tin epomeni epanalipsi

Edge next\_edge = graph->edge[i++]

int x = find(subsets, next\_edge.src)

int y = find(subsets, next\_edge.dest)

// an den dimiourgi kiklo tin vallo sto result

// kai avksano index tou result gia epomeni akmi

if (x != y) {

result[e++] = next\_edge

Union(subsets, x, y)

}

//allios aporripto tin epomeni akmi

}

// tipono to content tou result mu kai deixno to MST

int minimumCost = 0

for (i = 0; i < e; ++i)

minimumCost = minimumCost + result[i].weight

cout <<"MST ("<<minimumCost<<"):"<<endl

for (i = 0; i < e; ++i)

{

cout << result[i].src << " - " << result[i].dest

<< " (" << result[i].weight <<")"<< endl

} }